

# **THE FRAMEWORK FOR ENTERPRISE ARCHITECTURE: Background, Description and Utility**

by

**John A. Zachman**




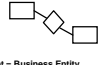
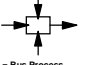
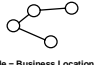
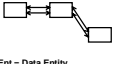
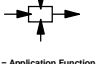
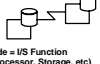

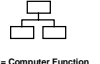




© Copyright 1996 Zachman International

In the early '80's, there was little interest in the idea of Enterprise Reengineering or Enterprise Modeling and the use of formalisms and models was generally limited to some aspects of application development within the Information Systems community. The subject of "architecture" was acknowledged at that time, however, there was little definition to support the concept. This lack of definition precipitated the initial investigation that ultimately resulted in the "Framework for Information Systems Architecture." Although from the outset, it was clear that it should have been referred to as a "Framework for *Enterprise* Architecture," that enlarged perspective could only now begin to be generally understood as a result of the relatively recent and increased, world-wide focus on Enterprise "engineering."

The Framework as it applies to Enterprises is simply a logical structure for classifying and organizing the descriptive representations of an Enterprise that are significant to the management of the Enterprise as well as to the development of the Enterprise's systems. It was derived from analogous structures that are found in the older disciplines of Architecture/Construction and Engineering/Manufacturing that classify and organize the design artifacts created over the process of designing and producing complex physical products (e.g. buildings or airplanes.)

The Framework graphic in its most simplistic form depicts the design artifacts that constitute the intersection between the roles in the design process, that is, OWNER, DESIGNER and BUILDER; and the product abstractions, that is, WHAT (material) it is made of, HOW (process) it works and WHERE (geometry) the components are, relative to one another. Empirically, in the older disciplines, some other "artifacts" were observable that were being used for scoping and for implementation purposes. These roles are somewhat arbitrarily labeled PLANNER and SUB-CONTRACTOR and are included in the Framework graphic that is commonly exhibited. The Framework, as it is applied to an Enterprise, depicting Enterprise design artifacts (models,) using Enterprise terminology appears below.
















ENTERPRISE ARCHITECTURE - A FRAMEWORK

	DATA <i>What</i>	FUNCTION <i>How</i>	NETWORK <i>Where</i>
OBJECTIVES/ SCOPE (CONTEXTUAL)	List of Things Important to the business 	List of Processes the Business Performs 	List of Locations in which the Business Operates 
<i>Planner</i>	ENTITY = Class of Business Thing	Process = Class of Business Process	Node = Major Business Location
BUSINESS MODEL (CONCEPTUAL)	e.g. Semantic Model 	e.g. Business Process Model 	e.g. Business Logistics System 
<i>Owner</i>	Ent = Business Entity Reln = Business Relationship	Proc = Bus Process IO = Bus Resources	Node = Business Location Link = Business Linkage
SYSTEM MODEL (LOGICAL)	e.g. Logical Data Model 	e.g. Application Architecture 	e.g. Distributed System Architecture 
<i>Designer</i>	Ent = Data Entity Reln = Data Relationship	Proc = Application Function IO = User Views	Node = I/S Function (Processor, Storage, etc) Link = Line Characteristics
TECHNOLOGY CONSTRAINED MODEL (PHYSICAL)	e.g. Physical Data Model 	e.g. System Design 	e.g. Technology Architecture 
<i>Builder</i>	Ent = Segment/Table/etc. Reln = Pointer/Key/etc.	Proc = Computer Function IO = Data Elements/Sets	Node = Hardware/Systems Software Link = Line Specifications
DETAILED REPRESENTATIONS (OUT-OF-CONTEXT)	e.g. Data Definition 	e.g. Program 	e.g. Network Architecture 
<i>Sub-Contractor</i>	Ent = Field Reln = Address	Proc = Language Statement IO = Control Block	Node = Address Link = Protocol
FUNCTIONING ENTERPRISE	e.g. DATA	e.g. FUNCTION	e.g. NETWORK

The older disciplines of Architecture and Manufacturing have accumulated considerable bodies of product knowledge through disciplined management of the “product definition” design artifacts. This has enabled enormous increases in product sophistication and the ability to manage high rates of product change over time. Similarly, disciplined production and management of “Enterprise definition” (i.e. the set of models identified in the Framework for Enterprise Architecture) should provide for an accumulation of a body of Enterprise knowledge to facilitate enormous increases in Enterprise sophistication and accommodation of high rates of Enterprise change over time.

From the very inception of the Framework, some other product abstractions were known to exist because it was obvious that in addition to WHAT, HOW and WHERE, a complete description would necessarily have to include the remaining primitive interrogatives: WHO, WHEN and WHY. These three additional interrogatives would be manifest as three additional columns of models that, in the case of Enterprises, would depict: WHO does what work, WHEN do things happen and WHY are various choices made. The state of the art in terms of modeling formalisms, as well as the inclination to devote energy to produce these additional artifacts is still somewhat limited, certainly in the case of Enterprises. Because experience in modeling is so limited, the examples of models for the cells in the “other three columns” are much more hypothetical and much less empirical. However hypothetical they may be, the remaining three columns of models appear below.

ENTERPRISE ARCHITECTURE  
THE "OTHER THREE COLUMNS"

PEOPLE	TIME	MOTIVATION	
List of Organizations Important to the Business 	List of Events Significant to the Business 	List of Business Goals/Strategies 	SCOPE  <i>Planner</i>
People = Major Organizations e.g. Work Flow Model 	Time = Major Business Event e.g. Master Schedule 	Ends/Mean = Major Bus Goals/Critical Success Factors e.g. Business Plan 	BUSINESS MODEL (CONCEPTUAL)  <i>Owner</i>
People = Organization Unit Work = Work Product e.g. Human Interface Architecture* 	Time = Business Event Cycle = Business Cycle e.g. Processing Structure 	End = Business Objective Means = Business Strategy e.g. Business Rule Model 	SYSTEM MODEL (LOGICAL)  <i>Designer</i>
People = Role Work = Deliverable e.g. Presentation Architecture 	Time = System Event Cycle = Processing Cycle e.g. Control Structure 	End = Structural Assertion Means = Action Assertion e.g. Rule Design 	TECHNOLOGY MODEL (PHYSICAL)  <i>Builder</i>
People = User Work = Screen Format e.g. Security Architecture 	Time = Execute Cycle = Component Cycle e.g. Timing Definition 	End = Condition Means = Action e.g. Rule Specification 	DETAILED REPRESENTATIONS (OUT-OF-CONTEXT)  <i>Sub-Contractor</i>
People = Identity Work = Job e.g. ORGANIZATION	Time = Interrupt Cycle = Machine Cycle e.g. SCHEDULE	End = Sub-condition Means = Step e.g. STRATEGY	FUNCTIONING ENTERPRISE

The Framework is a generic classification scheme for design artifacts, that is, descriptive representations of any complex object. The utility of such a classification scheme is to enable focused concentration on selected aspects of an object without losing a sense of the contextual, or holistic, perspective. In designing and building complex objects, there are simply too many details and relationships to consider simultaneously. However, at the same time, isolating single variables and making design decisions *out of context* results in sub-optimization with all its attendant costs and dissipation of energy. Restoration of integrity or retrofitting the sub-optimized components of the resultant object, such that they might approximate the purpose for which the object was originally intended, may well be financially prohibitive.

This is the condition in which many Enterprises find themselves after about fifty years of building automated systems, out-of-context. They have a large inventory of “current systems,” built out-of-context, not integrated, not supporting the Enterprise, that are consuming enormous amounts of resource for “maintenance” and are far and away too costly to replace. As a matter of fact, the inventory of existing systems has come to be referred to as “the legacy,” a kind-of “albatross,” a penalty to be paid for the mistakes of the past.

A balance between the holistic, contextual view and the pragmatic, implementation view can be facilitated by a Framework that has the characteristics of any good classification scheme, that is, it allows for abstractions intended to:

- a. simplify for understanding and communication, and
- b. clearly focus on independent variables for analytical purposes, but at the same time,

- c. maintain a disciplined awareness of contextual relationships that are significant to preserve the integrity of the object.

It makes little difference whether the object is physical, like an airplane, or conceptual, like an Enterprise. The challenges are the same. How do you design and build it piece-by-piece such that it achieves its purpose without dissipating its value and raising its cost by optimizing the pieces, sub-optimizing the object.

Although the Framework for Enterprise Architecture is an application of Framework concepts to Enterprises, the Framework itself is generic. It is a comprehensive, logical structure for descriptive representations (i.e. models, or design artifacts) of any complex object and is neutral with regard to the processes or tools used for producing the descriptions. For this reason, the Framework, as applied to Enterprises, is helpful for sorting out very complex, technology and methodology choices and issues that are significant both to general management and to technology management.

In summary, the Framework is:

- a. **SIMPLE** - it is easy to understand ... not technical, purely logical. In its most elemental form, it is three perspectives: Owner, Designer, Builder ... and three abstractions: Material, Function, Geometry. Anybody (technical or non-technical) can understand it.
- b. **COMPREHENSIVE** - it addresses the Enterprise in its entirety. Any issues can be mapped against it to understand where they fit within the context of the Enterprise as a whole.
- c. a **LANGUAGE** - it helps you think about complex concepts and communicate them precisely with few, non-technical words.
- d. a **PLANNING TOOL** - it helps you make better choices as you are never making choices in a vacuum. You can position issues in the context of the Enterprise and see a total range of alternatives.
- e. a **PROBLEM-SOLVING TOOL** - it enables you to work with abstractions, to simplify, to isolate simple variables without losing sense of the complexity of the Enterprise as a whole.
- f. **NEUTRAL** - it is defined totally independently of tools or methodologies and therefore any tool or any methodology can be mapped against it to understand their implicit trade-offs ... that is, what they are doing, and what they are NOT doing.

The Framework for Enterprise Architecture is not “the answer.” It is a tool ... a tool for thinking. If it is employed with understanding, it should be of great benefit to technical

and non-technical management alike in dealing with the complexities and dynamics of the Information Age Enterprise.

*References*

1. "A Framework for Information Systems Architecture." John A. Zachman. *IBM Systems Journal*, vol. 26, no. 3, 1987. IBM Publication G321-5298. 914-945-3836 or 914-945-2018 fax.
2. "Extending and Formalizing the Framework for Information Systems Architecture." J.F. Sowa and J. A. Zachman. *IBM Systems Journal*, vol. 31, no. 3, 1992. IBM Publication G321-5488. 1-818-945-3836.

*John A. Zachman  
Zachman International  
2222 Foothill Blvd. Suite 337  
La Cañada, Ca. 91011  
818-244-3763 (Phone and Fax)  
johnzachman@compuserve.com*